JAMES COOK UNIVERSITY

System Description: PTTP+GLiDeS Semantically Guided PTTP

Marianne Brown and Geoff Sutcliffe

00/01

SCHOOL OF INFORMATION TECHNOLOGY

TOWNSVILLE
QUEENSLAND 4811
AUSTRALIA

Title	System Description: PTTP+GLiDeS Semantically Guided PTTP
Primary Author(s)	Marianne Brown and Geoff Sutcliffe
Contact Information	School of Information Technology James Cook University {marianne,geoff}@cs.jcu.edu.au
Date	January 25, 2000

ISSN 1326-4524.

Abstract

PTTP+GLiDeS is a semantically guided linear deduction theorem prover, built from PTTP and MACE. It takes problems in clause normal form, generates semantic information about the clauses, and then uses the semantic information to guide its search for a proof. This paper outlines the semantic guidance strategy used by PTTP+GLiDeS and describes its implementation. The system's performance is evaluated against that of its parent system PTTP and its strengths and weaknesses are discussed.

Introduction

PTTP+GLiDeS is a semantically guided linear deduction theorem prover, built from PTTP [7] and MACE [5]. It takes problems in clause normal form (CNF), generates semantic information about the clauses, and then uses the semantic information to guide its search for a proof.

In the last decade there has been some work done in the area of semantic guidance, in a variety of first order theorem proving paradigms: SCOTT [6] is based on OTTER and is a forward chaining resolution system; CLIN-S [3] uses hyperlinking; RAMCS [2] uses constrained clauses to allow it to search for proofs and models simultaneously; and SGLD [9] is a chain format linear deduction system based on Graph Construction. Of these, CLIN-S and SGLD need to be supplied with semantics by the user. SCOTT uses FINDER [6] to generate models, and RAMCS generates its own models.

The Semantic Guidance

PTTP+GLiDeS uses a semantic pruning strategy that is based upon the strategy that can be applied to linear-input deductions. In a completed linear-input refutation, all centre clauses are FALSE in all models of the side clauses. This leads to a semantic pruning strategy that, at every stage of a linear-input deduction, requires all centre clauses in the deduction so far to be FALSE in a model of the side clauses. To implement this strategy it is necessary to know which are the potential side clauses, so that a model can be built. A simple possibility is to choose a negative top clause from a set of Horn clauses, in which case the mixed clauses are the potential side clauses. More sensitive analysis is also possible [4, 8]. Unfortunately, the extension of this pruning strategy to linear deduction is not direct. The possibility of ancestor resolutions means that centre clauses may be TRUE in a model of the side clauses.

In PTTP+GLiDeS, rather than placing a constraint on entire centre clauses, a semantic constraint is placed on certain literals of the centre clauses: The input clauses other than the chosen top clause of a linear deduction are named the *model clauses*. In a completed linear refutation, all centre clause literals that have resolved against input clause literals are required to be FALSE in a model of the model clauses. TRUE centre clause literals must be resolved against ancestor clause literals.

PTTP+GLiDeS implements linear deduction using the Model Elimination (ME) paradigm. In ME, A-literals are created by resolving the end B-literals of the centre chains with B-literals in side chains. PTTP+GLiDeS maintains a list of all the A-literals created throughout the entire deduction. This list is called the A-list. The pruning strategy requires that at every stage of the deduction, there must exist at least one ground instance of the A-list that is FALSE in a model of the model clauses. The result is that only FALSE B-literals are extended upon, and TRUE B-literals must reduce. Figure 1 shows an example of a PTTP+GLiDeS refutation. The problem clauses are $\{\sim money \vee tickets(buy), \sim tickets(sell) \vee money, money \vee tickets(X), \sim money \vee \sim tickets(X)\}$. The clause $\sim money \vee \sim tickets(X)$ is chosen to form the top chain, so that the other three clauses are the model clauses. The model M is $\{money, tickets(buy), \sim tickets(sell)\}$. The A-list is shown in braces under the centre chains.

Since the work described in [1], PTTP+GLiDeS has been enhanced to order the use of side chains, using the model of the model clauses. The model is used to give a score to each clause as follows: If there are N ground domain instances of a clause C with k literals, then for each literal L, n_L is the number of TRUE instances of L within the N ground instances. L is given the score $\frac{n_L}{N}$. The score for the clause C is $\frac{1}{N*k}\sum_{L=1}^{k} n_L$. The clause set is then ordered in descending order of scores. This gives preference to clauses that have more TRUE literal instances in the model. The use of these clauses leads to early pruning and forces the deduction into areas more likely to lead to a proof.

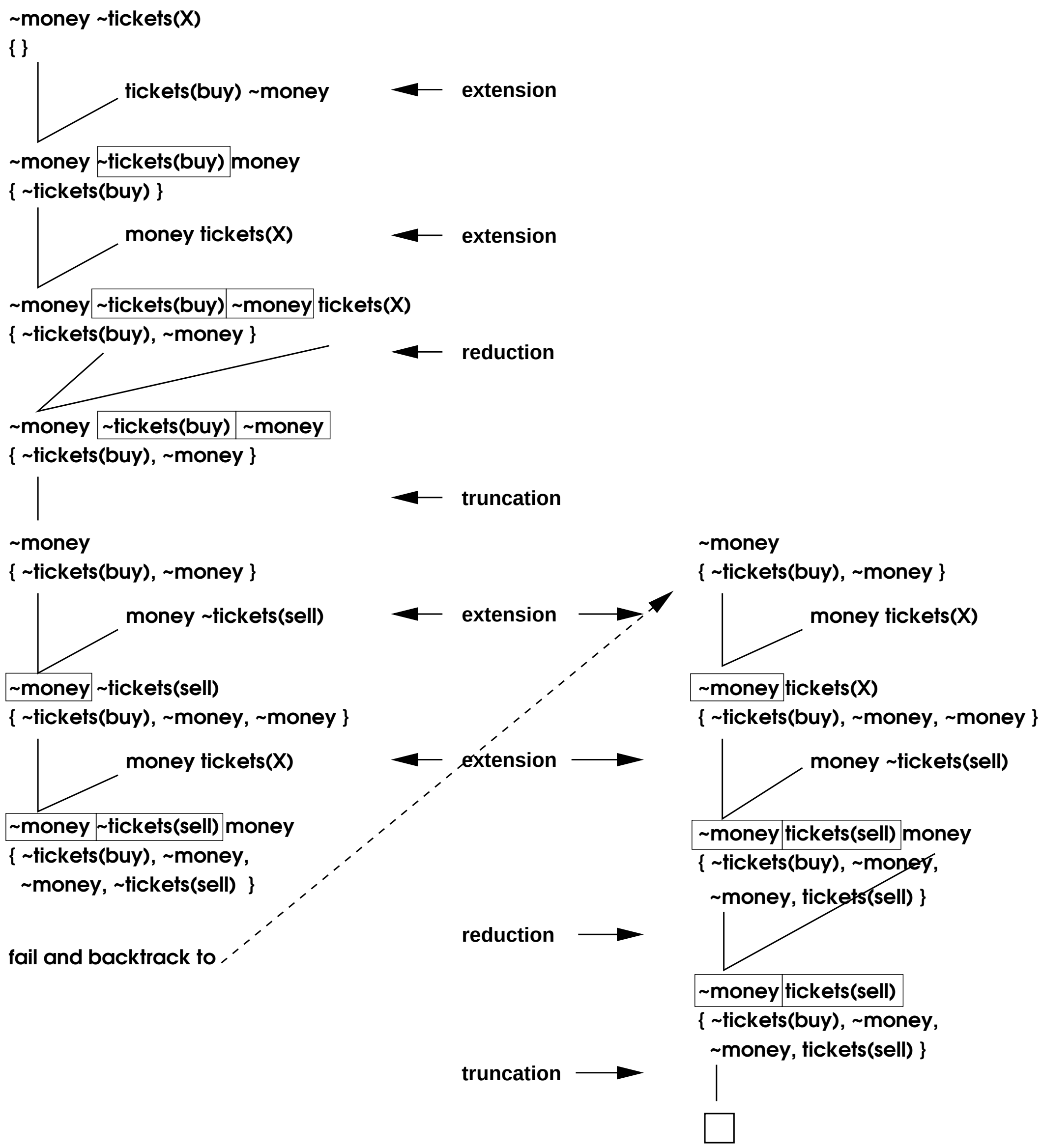

Figure 1: A PTTP+GLiDeS refutation

Implementation

PTTP+GLiDeS consists of a modified version of PTTP version 2e and MACE v1.3.3, combined with a `csh` script. It requires the problem to be presented in both PTTP and OTTER formats. The OTTER format file is processed so that it contains only the model clauses, and is used by MACE.

Initially the domain size for the model to be generated by MACE is set to equal the number of constants in the problem. If a model of this size cannot be found, the domain size is reset to 2 and MACE is allowed to determine the domain size. If no model is found PTTP+GLiDeS exits. If a model is found, the modified PTTP is started. The modified PTTP uses the model to reorder the clause set, then transforms the reordered clauses into Prolog procedures that implement the ME deduction and maintain the A-list. A semantic check is performed on the A-list after each extension and reduction operation. If the A-list does not have an instance in which every literal evaluates to FALSE in the model provided by MACE, then the extension or reduction is rejected.

Performance

Testing was carried out on 541 "difficult" problems from the TPTP problem library [10] v2.1.0. Both PTTP and PTTP+GLiDeS were tested on the same problems under the same conditions. Experiments where carried out on a SunSPARC 20 server using ECLiPSe v3.7.1 as the Prolog engine. A CPU time limit of 300 seconds was used. The results are summarized in Table 1. MACE failed to generate a model in 272 cases, and so PTTP+GLiDeS couldn't attempt those problems. Of those 269 problem where models were generated, worst performance is on Horn problems: all of the problems solved by PTTP and not PTTP+GLiDeS are Horn. MACE tends to generate trivial models (positive literals TRUE) for Horn problems. If the top centre clause is negative then, for a Horn clause set, a trivial model does not lead to any pruning. With the additional overhead of the semantic checking this leads to poor performance by PTTP+GLiDeS. Of the 269 models produced by MACE, 155 were effective, i.e., resulted in some pruning. Of the problems with effective models solved by both PTTP and PTTP+GLiDeS, in 13 out of 18 cases PTTP+GLiDeS had a lower inference count; in some cases significantly lower. This is shown by the fact that the average number of inferences for PTTP+GLiDeS is 2.5 times smaller than that of PTTP. This shows that the pruning is having a positive effect. PTTP+GLiDeS performs best on non-Horn problems. Table 2 shows some results where PTTP+GLiDeS performed better than PTTP. For these problems even trivial models can be of assistance.

Total number of problems:	541 (311/230)	(Horn/Non-Horn)
CPU time limit:	300 s	
Number of models generated:	269 (227/42)	
Number of problems solved from 269:	PTTP	PTTP+GLiDeS
	66 (60/6)	59 (51/8)
Number of effective models generated:	155 (120/35)	
Number of problems solved from 155:	PTTP	PTTP+GLiDeS
	21 (16/5)	20 (13/7)
For the 18 problems (from 155) solved by both systems:		
	PTTP	PTTP+GLiDeS
Average CPU Time:	34.24	69.18
Average Number of Inferences:	119634.28	47812.22
Average Number of Rej. Inferences:		3896.78

Table 1: Summary of experimental data.

Problem	Model	PTTP+GLiDeS CPU Time	PTTP+GLiDeS Inferences	PTTP+GLiDeS Rejected Inferences	PTTP CPU Time	PTTP Inferences
CAT003-3	non-Trivial	68.5	64232	10451	TIMEOUT	
CAT012-3	Trivial	32.0	49220	4174	54.6	175367
GRP008-1	Trivial	248.3	404198	3937	TIMEOUT	
SYN071-1	non-Trivial	70.1	84908	27653	262.8	832600

Table 2: Results for some non-Horn problems where PTTP+GLiDeS out-performs PTTP.

With respect to ordering of the clause set, experiments have been carried out using both ascending and descending with respect to the truth score. Initially it was thought that ordering the clause set in ascending order of truth score (from 'less TRUE' to 'more TRUE') would lead the search away from pruning and therefore towards the proof. This turns out not to be the case. While the results are not statistically significantly different in terms of rejected inferences and inferences, descending ordering solved 4 more problems overall, of which 3 had effective models. As solving problems is the most significant measure of a theorem prover's ability this shows that pruning early is more effective.

Conclusion

In those cases where a strongly effective model has been obtained, results are good. This leads to the question, "what makes a model effective?" At present the first model generated by MACE is used. If the characteristics of a strongly effective model can be quantified then it should be possible to generate many models and select the one most likely to give good performance.

On the completeness issue, this prover prunes away proofs which contain complementary A-literals on different branches of the tableau. In the few cases examined to date, another proof that conforms to this extended admissibility rule has always been found. Whether there is always another such proof is not known.

References

[1] M. Brown and G. Sutcliffe. PTTP+GLiDeS: Guiding Linear Deductions with Semantics. In N. Foo, editor, *Proceedings of AI'99: The 12th Australian Joint Conference on Artificial Intelligence*, number 1747 in LNAI, pages 244–254. Springer-Verlag, 1999.

[2] R. Caferra and N. Peltier. Extending Semantic Resolution via Automated Model Building: Applications. In C.S. Mellish, editor, *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 328–334. Morgan Kaufmann, 1995.

[3] H. Chu and D. Plaisted. Semantically Guided First-order Theorem Proving using Hyper-linking. In A. Bundy, editor, *Proceedings of the 12th International Conference on Automated Deduction*, number 814 in LNAI, pages 192–206. Springer-Verlag, 1994.

[4] D.A. de Waal and J.P. Gallagher. The Applicability of Logic Programming Analysis and Transformation to Theorem Proving. In A. Bundy, editor, *Proceedings of the 12th International Conference on Automated Deduction*, number 814 in LNAI, pages 207–221. Springer-Verlag, 1994.

[5] W.W. McCune. A Davis-Putnam Program and its Application to Finite First-Order Model Search: Quasigroup Existence Problems. Technical Report Technical Report ANL/MCS-TM-194, Argonne National Laboratory, Argonne, USA, 1994.

[6] J.K. Slaney. SCOTT: A Model-Guided Theorem Prover. In R. Bajcsy, editor, *Proceedings of the 13th International Conference on Artificial Intelligence*, pages 109–114. Morgan-Kaufman, 1993.

[7] M.E. Stickel. A Prolog Technology Theorem Prover: A New Exposition and Implementation in Prolog. Technical Report Technical Note 464, SRI International, Menlo Park, USA, 1989.

[8] G. Sutcliffe. Linear-Input Subset Analysis. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction*, number 607 in LNAI, pages 268–280, Saratoga Springs, NY, USA, June 1992. Springer-Verlag.

[9] G. Sutcliffe. The Semantically Guided Linear Deduction System. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction*, number 607 in LNAI, pages 677–680, Saratoga Springs, NY, USA, June 1992. Springer-Verlag.

[10] G. Sutcliffe and C.B. Suttner. The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.